Plánování a rozvrhování

7

Roman Barták, KTIML

roman.bartak@mff.cuni.cz
http://ktiml.mff.cuni.cz/~bartak

Čas a plánování

Konceptuální model plánování pracuje s **implicitním časem**:

- akce a události jsou instantní (nemají trvání)
- splnění cílů je požadováno jen na konci

Tento omezený pohled se hodí pro studium logiky a formální složitosti plánování.

- **Realita** je ale trochu **jiná**:
  - akce zabírají jistý **časový úsek**
  - **předpoklady** akce nemusí být vyžadovány jen při startu akce, ale i v době jejího trvání
  - **efekty** mohou začít platit před koncem akce, mohou být omezeny jen na dobu trvání akce nebo jsou opožděny
  - **efekty** akcí se mohou **sdružovat**
  - mohou se vyskytovat **průběžné cíle**

Modely času

Základní principy

Co je to čas?

Matematická struktura pro čas je obecně **množina s tranzitivní a asymetrickou relací uspořádání**.

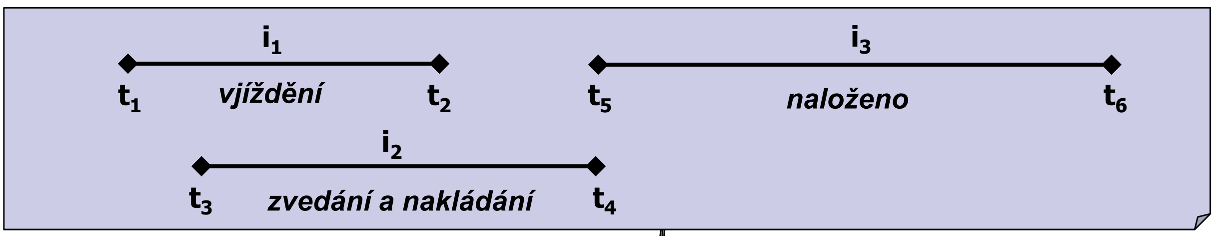
Množina může být spojitá (pro nás R) nebo diskrétní (pro nás Z).

Plánovací systém bude používat **databázi časových údajů**, kterou bude v průběhu plánování **updatovat** a bude se dotazovat na její **konzistenci**.

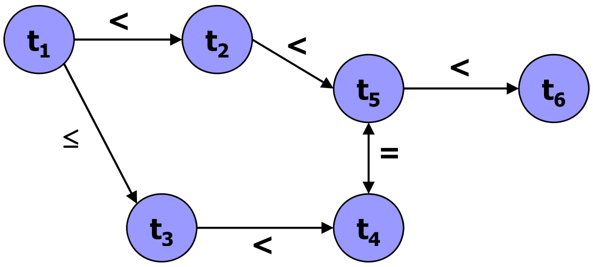
S časem můžeme pracovat:

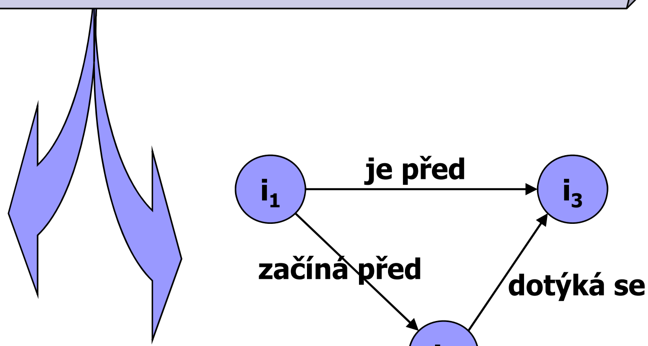
- **kvalitativně**
  - zajímají nás relativní vztahy (akce A skončila dříve než začala akce B)
- **kvantitativně**
  - zajímají nás absolutní hodnoty (akce A začala v čase 0 a skončila v čase 23)

- Robot začal vjíždět do zóny nakládání v čase t_1 a zastavil se zde v čase t_2.
- Jeřáb začal zvedat kontejner v čase t_3 a ukončil nakládání v čase t_4.
- V čase t_5 byl kontejner umístěn na robota a setrval zde až do času t_6.



Možné sítě podmínek:





Při **modelování času** nás budou zajímat:

- **časové reference**
  (kdy se něco stalo a kdy něco platilo)
  - **body změny** (okamžiky) kdy stavová proměnná nabyla jiné hodnoty
    okamžik je proměnná nad reálnými čísly
  - **časové periody** (intervaly) během kterých dané tvrzení platí
    interval je dvojice proměnných (x,y) nad reálnými čísly taková, že x<y
- **relativní vztahy** mezi časovými referencemi
  - **uspořádání** časových referencí

Algebra okamžiků
základy

- **Symbolický kalkulus pro práci s okamžiky.**
- Dva okamžiky t_1 a t_2 můžeme svázat pouze následujícími třemi způsoby:
  - $[t_1 < t_2]$
  - $[t_1 > t_2]$
  - $[t_1 = t_2]$

  Relace P = {<,=,>} budeme nazývat **primitivní vztahy**.
- Primitivní vztahy můžeme spojit do množiny a vyjadřovat tak libovolný (i méně přesný) vztah mezi okamžiky (množina primitivních vztahů vyjadřuje jejich disjunkci):
  - {}, {<}, {=}, {>}, {<,=}, {>,=}, {<,>}, {<,=,>}
- **vztah** r mezi dvěma časovými body t a t�much budeme zapisovat **[t r t‘]**
- Algebra okamžiků nám umožní **pracovat se vztahy symbolicky** bez nutnosti umístit okamžik do přesného času.

Algebra okamžiků
operace

- Máme množinu vztahů R:
  - {{}, {<}, {=}, {>}, {<,=}, {>,=}, {<,>}, {<,=,>}}
- Operace nad R:
  - **množinové operace** $\cap$, $\cup$
    - slouží pro vyjádření konjunkce a disjunkce vztahů
  - **operace • pro tranzitivitu**
    - pro odvození nového vztahu na základě existujících vztahů
    - $[t_1\ r\ t_2]$ a $[t_2\ q\ t_3]$ dává $[t_1\ r{\bullet}q\ t_3]$ podle tabulky

•	<	=	>
<	<	<	P
=	<	=	>
>	P	>	>

- Nejužitečnější budou operace $\cap$ a •, které umožní kombinovat existující vztahy a vztahy odvozené z tranzitivity:
  - $[t_1\ r\ t_2]$ a $[t_1\ q\ t_3]$ a $[t_3\ s\ t_2]$ dává $[t_1\ r \cap (q \bullet s)\ t_2]$

Algebra okamžiků
konzistence

- Množina okamžiků X společně s množinou (binárních) vztahů $r_{i,j} \in R$ nad těmito okamžiky C tvoří **síť podmínek** (X,C).
  - Pokud není v C explicitně uveden vztah mezi dvěma okamžiky (i,j), bereme univerzální vztah P.
- Říkáme, že síť podmínek (X,C) je **konzistentní**, jestliže je možné každému okamžiku přiřadit reálné číslo tak, že všechny vztahy jsou splněny (existuje řešení).

Tvrzení:

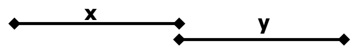
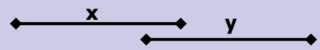
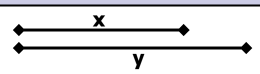
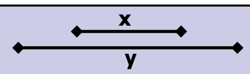
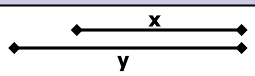
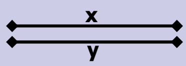
Síť podmínek (X,C) je konzistentní, právě když existuje množina primitivních vztahů $p_{i,j} \in r_{i,j}$ takových, že pro libovolnou trojici těchto vztahů platí $p_{i,j} \in p_{i,k} \bullet p_{k,j}$.

Poznámky:

Pro zjištění konzistence sítě stačí udělat její tranzitivní uzávěr např. pomocí algoritmu konzistence po cestě (PC).
- Pokud někde dostaneme vztah {}, potom síť není konzistentní.
- V opačném případě je síť konzistentní (ale PC nemusí odstranit všechny primitivní vztahy, které jsou nekonzistentní).

Algebra intervalů
základy

- **Symbolický kalkulus pro práci s intervaly** (interval je dán dvojicí okamžiků i^- a i^+ takových, že $[i^- < i^+]$).
- Primitivní vztahy mezi intervaly:

x **b**efore y	$x^+ < y^-$	
x **m**eets y	$x^+ = y^-$	
x **o**verlaps y	$x^- < y^- < x^+ \wedge x^+ < y^+$	
x **s**tarts y	$x^- = y^- \wedge x^+ < y^+$	
x **d**uring y	$y^- < x^- \wedge x^+ < y^+$	
x **f**inishes y	$y^- < x^- \wedge x^+ = y^+$	
x **e**quals y	$x^- = y^- \wedge x^+ = y^+$	
b′, m′, o′, s′, d′, f′	symetrické vztahy	

Algebra intervalů
operace a konzistence

- Primitivní vztahy opět můžeme kombinovat v množinách (2^{13} vztahů).
  - □ Někdy se vybírá jen podmnožina možných vztahů, které mají v dané aplikaci nějaký význam.
    - např. {b,m,b′,m′} znamená nepřekrývání a hodí se pro modelování unárních zdrojů
- množinové operace $\cap$, $\cup$ a operace • pro tranzitivitu
- Říkáme, že síť podmínek s intervaly (X,C) je **konzistentní**, jestliže existují reálná čísla x_i^-, x_i^+ popisující každý interval x_i tak, že všechny vztahy jsou splněny.

Tvrzení:

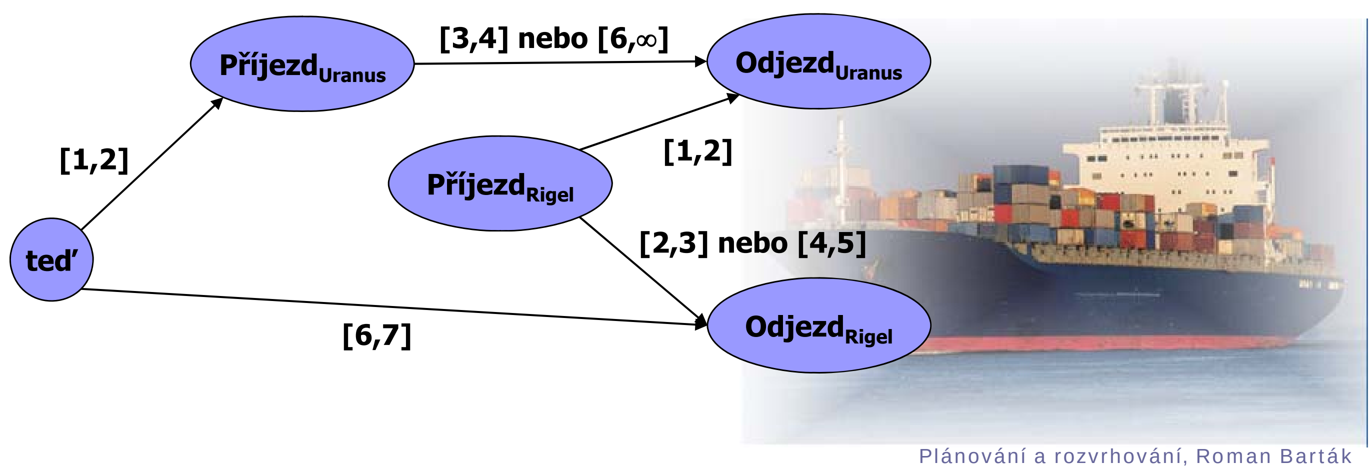
Síť podmínek s intervaly (X,C) je konzistentní, právě když existuje množina primitivních vztahů $p_{i,j} \in r_{i,j}$ takových, že pro libovolnou trojici těchto vztahů platí $p_{i,j} \in p_{i,k} \bullet p_{k,j}$.

Poznámky:

- □ Konzistence po cestě není pro sítě podmínek s intervaly úplná konzistenční technika! Může ale detekovat některé nekonzistence.
- □ Problém konzistence sítě podmínek s intervaly je NP-úplný.
- □ Intervaly lze převést na okamžiky, ale některé podmínky pak nebudou binární.

Kvantitativní přístup
příklad

- Do přístavu míří dvě lodě, Uranus a Rigel.
- Uranus přijede za jeden až dva dny.
- Uranus bude odjíždět buď s lehkým nákladem, pak stráví v přístavu tři až čtyři dny, nebo odjede plně naložený a pak stráví v přístavu minimálně šest dní.
- Rigel bude obsloužen buď v expresním doku, kde stráví dva až tři dny, nebo v normálním doku, kde stráví čtyři až pět dní.
- Uranus musí odjet jeden až dva dny po příjezdu Rigelu.
- Rigel musí odjet šest až sedm dní ode dneška.



Kvantitativní přístup
formálně

- Základními primitivy jsou opět **okamžiky**, tentokrát ale s numericky vyjádřenými vztahy.
- Jednoduché **časové podmínky** pro okamžiky t_i a t_j:
  - unární: $a_i \leq t_i \leq b_i$
  - binární: $a_{ij} \leq t_i - t_j \leq b_{ij}$,

  a_i, b_i, a_{ij}, b_{ij} jsou (reálné) konstanty

Poznámky:

- Unární vztah můžeme převést na binární, pokud použijeme nějaký pevný referenční bod t_0.
- Podmínku mezi okamžiky t_i a t_j budeme zapisovat $[a_{ij},b_{ij}]$.
- Můžeme pracovat s disjunkcemi jednoduchých časových podmínek.

STN

- **Simple Temporal Network (STN)**
  - pracujeme pouze s jednoduchými časovými podmínkami $r_{ij} = [a_{ij},b_{ij}]$
  - **operace**:
    - skládání podmínek: $r_{ij} \bullet r_{jk} = [a_{ij}+a_{jk}, b_{ij}+b_{jk}]$
    - průnik podmínek: $r_{ij} \cap r'_{ij} = [\max\{a_{ij},a'_{ij}\}, \min\{b_{ij},b'_{ij}\}]$
  - **STN je konzistentní** pokud existuje přiřazení hodnot do okamžiků takové, že všechny podmínky jsou splněny.
  - Konzistenci STN lze zařídit jednoprůchodovým algoritmem pro **konzistenci po cestě** (dokonce odfiltruje všechny nekonzistence) nebo známým **Floyd-Warshallovým algoritmem** pro hledání minimálních vzdáleností mezi každou dvojicí bodů.

Algoritmy

- **Konzistence po cestě**
  - v podstatě počítá tranzitivní uzávěr binárních relací r
  - pro STN stačí jeden průchod (obecně algoritmus iteruje dokud dochází ke změnám)
  - může fungovat inkrementálně

jednoprůchodový pro STN

```
PC(X, C)
    for each k : 1 ≤ k ≤ n do
        for each pair i,j : 1 ≤ i < j ≤ n,, i ≠ k, j ≠ k do
            r_ij ← r_ij ∩ [r_ik • r_kj]
            if r_ij = Ø then exit(inconsistent)
end
```

obecný

```
PC(C)
    until stabilization of all constraints in C do
        for each k : 1 ≤ k ≤ n do
            for each pair i,j : 1 ≤ i < j ≤ n, i ≠ k, j ≠ k do
                c_ij ← c_ij ∩ [c_ik • c_kj]
                if c_ij = Ø then exit(inconsistent)
end
```

- **Floyd-Warshallův algoritmus**
  - počítá nejkratší vzdálenost mezi každou dvojicí bodů
  - síť se musí nejprve upravit
    - vrcholy i a j spojuje hrana se vzdáleností b_{ij}
    - vrcholy j a i spojuje hrana se vzdáleností $-a_{ij}$.
  - STN je konzistentní, právě když v grafu nejsou negativní cykly, tj. $d(i,i) \geq 0$

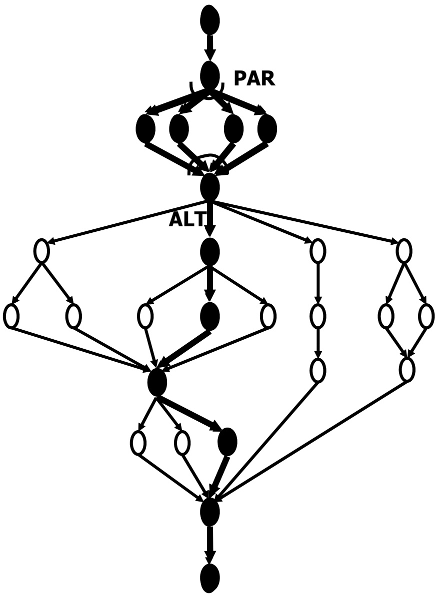
```
Floyd-Warshall(X, E)
    for each i and j in X do
        if (i,j) ∈ E then d(i,j) ← l_ij else d(i,j) ← ∞
        d(i,i) ← 0
    for each i,j,k in X do
        d(i,j) ← min{d(i,j), d(i,k) + d(k,j)}
end
```


TCSP

- **Temporal Constraint Network (TCSP)**
  - Povolíme **disjunkce** jednoduchých časových podmínek.
  - Operace • a ∩ se musí provádět s množinami intervalů.
  - **TCSP je konzistentní** pokud existuje přiřazení hodnot do okamžiků takové, že všechny podmínky jsou splněny.
  - Algoritmus konzistence po cestě zde není úplný (negarantuje konzistenci)!
  - Jednoduchý **řešící přístup**:
    - síť podmínek se rozloží na několik STN (pro každou disjunkci se zvolí jedna část)
    - odděleně se vyřeší získané STN
    - výsledek se získá sjednocením řešení jednotlivých STN

TNA

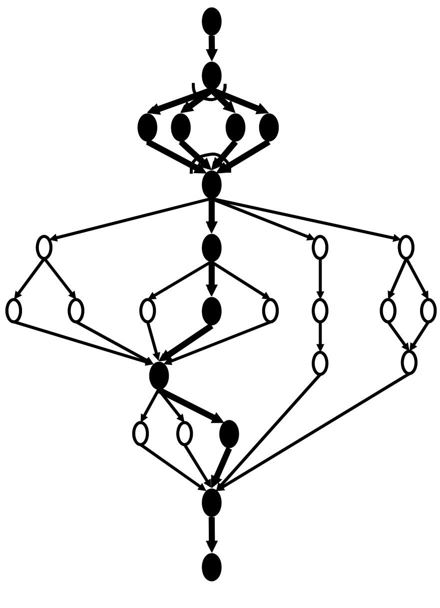
- **Temporální síť s alternativami** (TNA) je orientovaný graf se speciálním tvarem logických závislostí mezi uzly – **relace větvení**.



  - Proces se může rozdělit na **paralelní větve**, tj. uzly paralelních větví budou zpracovány paralelně (všechny uzly z větví musí být přítomny v řešení).
  - Proces může také volit mezi **alternativními větvemi**, tj. zpracují se uzly právě jedné větve (pouze uzly z vybrané větve budou přítomny v řešení).
  - Otázkou je, **které uzly jsou validní** v řešení (jsou přítomny ve vybraném pod-grafu, který splňuje logické i temporální podmínky).

Složitost problému

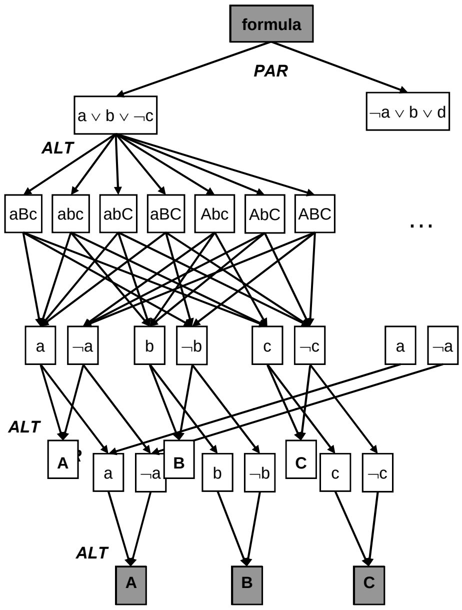
- Pokud uděláme všechny uzly nevalidní (všechny uzly jsou z grafu odstraněny), potom získáme **triviální řešení** splňující všechny podmínky.



  - Předpokládejme nyní, že některé předem **vybrané uzly musí být validní**, tj. jsou určeny pro zahrnutí do hledaného pod-grafu.
    - například, uzel odpovídající objednávce musí být přítomen
  - Otázkou je, jestli je potom těžké rozhodnout, zda lze **vybrat pod-graf obsahující zvolené uzly tak, aby byly splněny podmínky větvení.**
    - Jinými slovy, je možné vybrat proces tak, aby byla splněna objednávka?

Je to těžký problém!

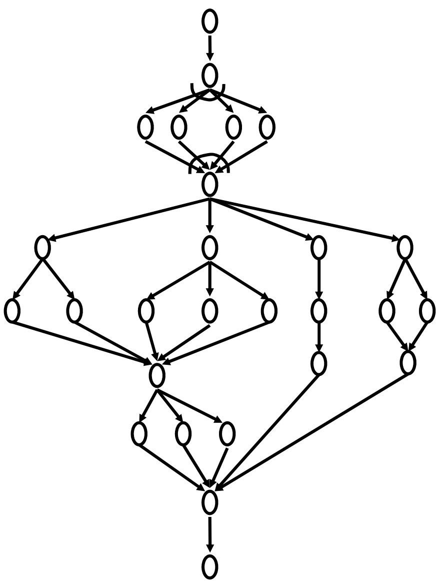
- Problém rozhodnutí o validitě/ne-validitě uzlů v P/A grafu je **NP-úplný**.



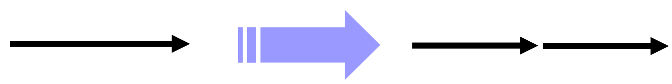
- převedeme na něj **problém 3SAT**
- **kódujeme klauzuli** jako P/A graf tak, že klauzule je splněna právě když existuje pod-graf splňující větvící podmínky
  - příklad (a ∨ b ∨ ¬c)
- přidáme uzly pro proměnné a jejich hodnoty a uzel pro formuli
- nastavíme **validní uzly**

Reálné procesy

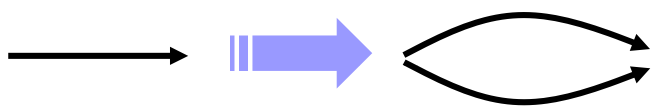
- Reálné výrobní procesy mají často velmi **specifickou strukturu**.



- Procesní síť často vzniká postupnou **dekompozicí** „meta-procesů“ na specifické pod-procesy/operace:
  - **sériová dekompozice**

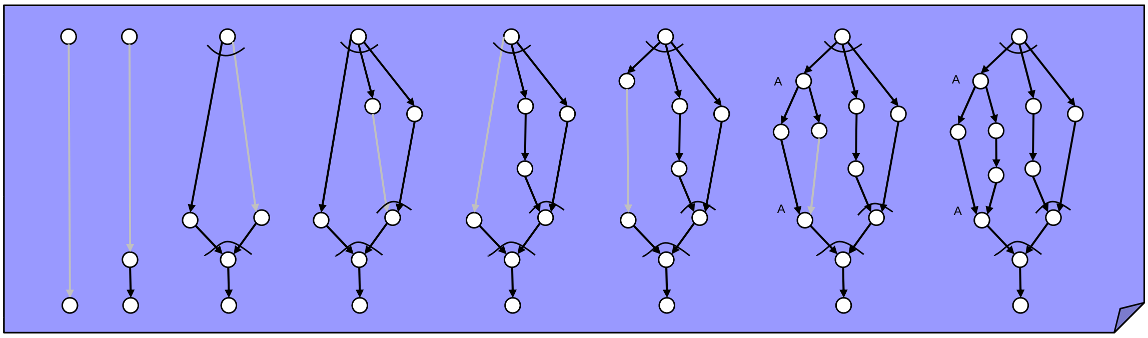


  - **paralelní/alternativní dekompozice**



Zahnízděné grafy

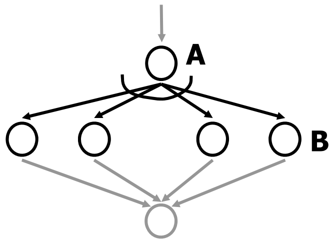
- grafy zkonstruované z jedné hrany postupnou aplikací následující **dekompoziční operace**:

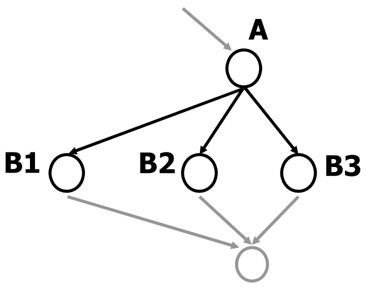


- **Vlastnosti:**
  - jedná se o temporální síť s alternativami
  - zahnízděné grafy lze algoritmicky rozpoznat
  - problém výběru validních uzlů je polynomiálně řešitelný

Logické relace

- Problém výběru validních uzlů lze popsat jako **problém splňování omezujících podmínek.**





  - každý **uzel** A je popsán proměnou V_A s doménou {0,1}
  - každá hrana (A,B) z **paralelního větvení** definuje podmínku $V_A = V_B$
  - nechť (A,B1),…, (A,Bk) jsou všechny hrany nějakého **alternativního větvení**, potom lze definovat podmínku $V_A = \Sigma_{i=1,\dots,k} V_{Bi}$

- Popsaný základní model **nedosahuje globální konzistence** (A = 1, A = B+C, D = B+C $\not\Rightarrow$ D = 1), ale pro zahnízděné sítě lze získat globální konzistenci.

Temporální relace

- Zatím jsme předpokládali, že hrana v grafu představuje **precedenci** (jeden uzel předchází před druhým).
- Každou hranu (X,Y) můžeme také označit **jednoduchou temporální podmínkou** [a,b] s významem $\mathbf{a \le Y\text{-}X \le b}$.
  - **(Zahnízděné) Temporální sítě s alternativami**
- Základní CSP model:
  - každý **uzel** A je popsán **temporální proměnnou** T_A s doménou ⟨0,MaxTime⟩, kde MaxTime je uživatelem zadaná konstanta.
  - Temporální relace [a,b] mezi uzly X a Y musí platit pokud jsou oba uzly validní!

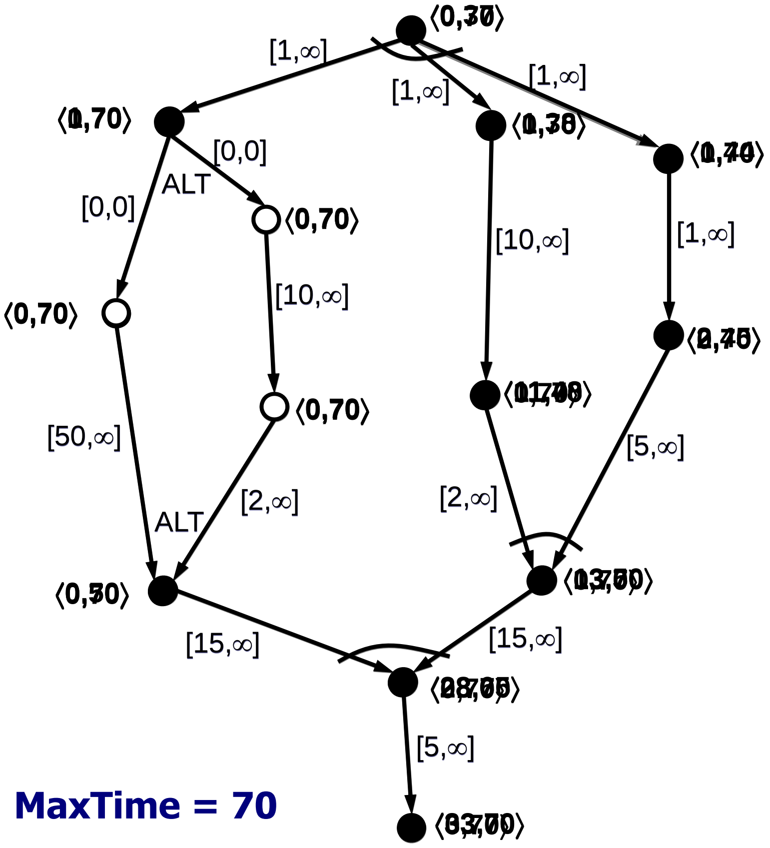
$$\mathbf{V_X * V_Y * (T_X + a) \le T_Y \ \wedge V_X * V_Y * (T_Y - b) \le T_X.}$$

Poznámky:

- $V_X = 0 \vee V_Y = 0 \rightarrow 0 \le T_Y \ \wedge 0 \le T_X$
- $V_X = V_Y = 1 \rightarrow (T_X + a) \le T_Y \ \wedge (T_Y - b) \le T_X$.
- Zmíněné temporální podmínky neuvažují typ větvení!

Slabá filtrace

temporálních relací



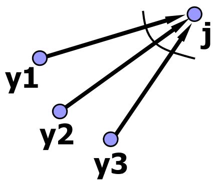
- Základní model opět nedosahuje globální konzistence!
  - Nejlevnější uzel nemůže být validní kvůli temporálním podmínkám.
- Sílu filtrace ale můžeme zvýšit následujícími způsoby:
  - temporální podmínky budou vždy filtrovat,
  - budeme integrovat logické a temporální uvažování.

Temporální filtrace

Silnější filtrace je založena na dvou myšlenkách:

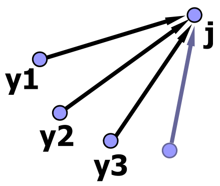
- **vždy filtrujeme** temporální podmínku (dokud nějaká doména nebude prázdná)
- při temporální filtraci **uvažujeme typ větvení**

- Propagace „po proudu“ (proti proudu obdobně)
  - **paralelní větvení**



  - $d(t_j) \leftarrow d(t_j) \cap \bigcap_{i=1,\ldots,k}\{ (d(t_{yi}) + \langle a_{yi,x}, b_{yi,x}\rangle) \}$ je-li neprázdné
  - $d(v_j) \leftarrow d(v_j) \cap \{0\}$ pokud se doména temporální prom. vyprázdní

  - **alternativní větvení** (konstruktivní disjunkce)



  - $d(t_j) \leftarrow d(t_j) \cap \bigcup_{i=1,\ldots,k}\{ (d(t_{yi}) + \langle a_{yi,x}, b_{yi,x}\rangle) \}$ je-li neprázdné
  - $d(v_j) \leftarrow d(v_j) \cap \{0\}$ pokud se doména vyprázdní

Složitost problému

- **Je možné polynomiálně získat globální konzistenci temporálních podmínek v zahnízděných sítích?**
- Bohužel, jedná se o **NP-úplný problém** ☹
  - Na problém konzistence temporálních podmínek lze převést **subset sum problém**.
  - Nechť Z_i, i = 1,…,n jsou celá čísla. Existuje podmnožina S z {1,…,n} taková, že $\sum_{i\in S} Z_i = K$?